PARALLEL PROCESSING OF ELECTROMAGNETIC AND MAGNETIC DATA SETS FOR UXO DETECTION.

Neil Dunstan
University of New England
Armidale, NSW, 2351, AUSTRALIA
Email: neil@cs.une.edu.au
Tel: 61 2 67732119 I Fax: 61 2 67733312

Peter Clark
Geophysical Technology Limited
PO Box U9, Armidale, NSW, 2351, AUSTRALIA
Email: clarkpj@ozemail.com.au
Tel: 61 2 67733508 Fax: 61 2 67733307

Category: UXO Detection, Technologies Session

Abstract

The simultaneous measurement of magnetic and electromagnetic data sets for unexploded ordnance (UXO) span the same spatial frequency response at sub-metre intervals. These data sets are ideal for the simultaneous, automatic detection and classification of UXO.

It is now common for computer equipment to include dedicated chips to accelerate the processing of high volume data sets. Such chips are often referred to as Application Specific Integrated Circuits (ASICs) and are hardwired for some fixed operation. Reconfigurable Computing is a maturing technology that allows hardware devices to be reconfigured for different operations, either between applications or dynamically during execution of some task. Reconfigurable Computing is possible using off the shelf devices called Field Programmable Gate Arrays (FPGAs) such as those manufactured by Altera and Xilinx.
A pipelined reconfigurable hardware device is proposed to support fast processing of UXO data sets. The design will be based on a systolic array design for two-dimensional convolution. Reconfigurable hardware is appropriate since the FPGA can be reconfigured for the detection of different objects at different orientations and depths.

A large database of magnetic and electromagnetic responses from UXO and common non-UXO items is available for testing detection and classifications. Initially, the development will be done as post-processing. During the next stage of development, dedicated hardware will be prototyped with the goal of detection and classification in real-time.

1. Introduction

Geophysical analysis commonly makes use of magnetic and electromagnetic sensor technologies to detect and classify subsurface objects. A major purpose of such systems is to support the clean up of areas made unsafe due to military activity. By analysing and comparing the data sets, the presence of subsurface objects, such as bombs, shells and landmines, can be detected. The signal responses constitute a large volume of data that takes considerable time to process. Processing entails analysis to detect anomalies that could indicate the presence of subsurface objects. Processing of data is done off-line and can take many hours to complete. As an example of the scale of the problem, Geophysical Technology

Limited (GTL) recently scanned a 200 acre site in the USA at sub-metre intervals (Clark, 1998). Dedicated, special-purpose chips are often used to accelerate the processing of high volume data sets. Such chips are often referred to as Application Specific Integrated Circuits (ASICs) and are hardwired for some fixed operation. Reconfigurable (or Custom) Computing is a maturing technology that allows hardware devices to be reconfigured for different operations, either between applications or dynamically during execution of some task. Reconfigurable hardware is available as off-the-shelf devices called Field Programmable Gate Arrays (FPGAs) such as those manufactured by Altera and Xilinx. FPGAs have traditionally been used for ``glue logic" and for prototyping ASICs. Current generation FPGAs contain thousands of reconfigurable logic cells. Petersen and Hutchings (1995) compared filtering applications in FPGAs, Digital Signal Processors (DSPs) and ASICs and concluded that FPGAs has superior performance to DSPs and comparable performance to ASICs. The feasibility of using FPGAs for a range of image processing tasks has been shown by Athnas and Abbott (1995) who developed a real-time image processing system using the Splash-2 Custom Computing Platform (Arnold et. al, 1992). The Splash-2's reconfigurable hardware consists of many boards each having sixteen FPGA chips. This paper investigates the use of FPGAs for object detection in magnetic and electromagnetic signal response data sets. The goal is to develop FPGA-based hardware accelerators to support host systems in UXO detection.

2. Geophysical Data Sets and Analysis

Each signal response data set is in the form of a two-dimensional array of values (16-bit values are assumed, although the designs that follow may be scaled for different sized values) representing the response at each point on a spatial grid. A data set X^{ij} can therefore be represented as shown in Figure 1 with superscripts distinguishing row and column coordinates of the spatial domain.

$$\begin{array}{ccccc}
X^{0,0} & X^{0,1} & X^{0,2} & \dots & X^{0,J-1} \\
X^{1,0} & X^{1,1} & X^{1,2} & \dots & X^{1,J-1} \\
X^{2,0} & X^{2,1} & X^{2,2} & \dots & X^{2,J-1} \\
\dots & \dots & \dots & \dots & \dots \\
X^{I-1,0} & X^{I-1,1} & X^{I-1,2} & \dots & X^{I-1,J-1}
\end{array}$$

Figure 1. A data set as a 2D array of values

Though the target subsurface object will often be three-dimensional it may also be characterised as a two dimensional signature response to a signal. That is, an object signature may be represented as a two-dimensional template of values H^{kl} as shown in Figure 2.

$$\begin{array}{cccc}
H^{0,0} & H^{0,1} & \dots & H^{0,L-1} \\
H^{1,0} & H^{1,1} & \dots & H^{1,L-1} \\
\dots & \dots & \dots & \dots \\
H^{K-1,0} & H^{K-1,1} & \dots & H^{K-1,L-1}
\end{array}$$

Figure 2. Signal response Signature as a 2D template

Furthermore, the signature response for the same object may be different for different depths and orientation of the object, requiring an object profile consisting of a collection of signatures. In order to detect and/or classify the object, it may be necessary to compare responses from different data sets, that is, types of signals. An additional complication is noise that may be introduced into the data sets from geological features in the ground and the sensing instrument itself. Object recognition therefore requires extensive filtering, analysis, and correlation of data sets and object profiles. For example, convolving one data set of *I ´ J* values with one object signature of *K ´ L* values requires in the order of *I ´ J ´ K ´ L c*omputations. Convolution is defined as

$$Y^{i,j} = \sum_{k=0}^{K-1}\sum_{l=0}^{L-1} H^{k,l} \times X^{i+k,j+l}, \quad 0 \le i \le I-1, \quad 0 \le j \le J-1 \quad 2.1$$

Significant speedup can be achieved if data analysis can be computed with parallel processing techniques.

3. Pipelined Parallelism

Pipelining refers to the breaking down of an operation into stages. Each stage is independent and can therefore execute at the same time as the others. It is therefore possible to have a number of operations under execution at the same time although at any particular time each one is at a different stage of execution. Figure 3 illustrates pipeline processing with a four--stage pipeline. Note: each operation goes through all four stages to be completed.

Stage 4				Op1	Op2	Op3
Stage 3			Op1	Op2	Op3	Op4
Stage 2		Op1	Op2	Op3	Op4	Op5
Stage 1	Op1	Op2	Op3	Op4	Op5	Op6
						Time →

Figure 3. A 4-Stage Pipeline. ***Opn*** **is part of operation** ***n***

The execution time for each operation remains the same as for sequential processing since each operation must undergo each of the stages one after another. The time taken for the execution of a complete operation is known as *latency*. The parallelism inherent in pipeline processing means that successive operations are being completed at a faster rate than if stages were not executed simultaneously. The rate at which successive operations are completed is known as *throughput*. Assuming that each of the *s* stages has the same duration of *d* seconds, then the latency is *s ´ d* seconds and the throughput is *1/d* operations per second. This implies a theoretical speedup of *s* over sequential processing.

One architectural form for pipeline processing is Systolic Arrays (Kung 1982) which are networks of processing cells through which data propagates in a synchronous fashion. Intermediate results are computed in the cells and final results emerge at the boundaries of the array. A systolic array design is shown in Figure 4.

4. Object Detection

Some correlation techniques for object detection are:

- *Sum of the absolute differences*. That is,

$$Y^{i,j} = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} | H^{k,l} - X^{i+k,j+l} |, \quad 0 \le i \le I-1, \quad 0 \le j \le J-1 \quad 4.1$$

- *Sum of the square of the differences*. That is,

$$Y^{i,j} = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} (H^{k,l} - X^{i+k,j+l})^2, \quad 0 \le i \le I-1, \quad 0 \le j \le J-1 \quad 4.2$$

- *Normalized correlation*. That is,

$$Y^{i,j} = \frac{(\sum_{k=0}^{K-1} \sum_{l=0}^{L-1} H^{k,l} \times X^{i+k,j+l})^2}{\sum_{k=0}^{K-1} \sum_{l=0}^{L-1} (X^{i+k,j+l})^2}, \quad 0 \le i \le I-1, \quad 0 \le j \le J-1 \quad 4.3$$

For the first two methods results close to zero indicate the best matches. For normalized correlation higher results imply better correlation. Note that convolution (equation 2.1) forms part of the numerator of normalized correlation.

5. Systolic Filters

In (Dunstan and Lenders, 1996) a novel systolic array design is presented for two-dimensional convolution featuring homogeneous processing cells. The array is in the shape of the template with each processing cell containing the corresponding template weight *H*. Rows of the image *X* enter the array and are processed by the processing cells. To process the entire image, each group of rows must be entered. The processor array is illustrated in Figure 4.

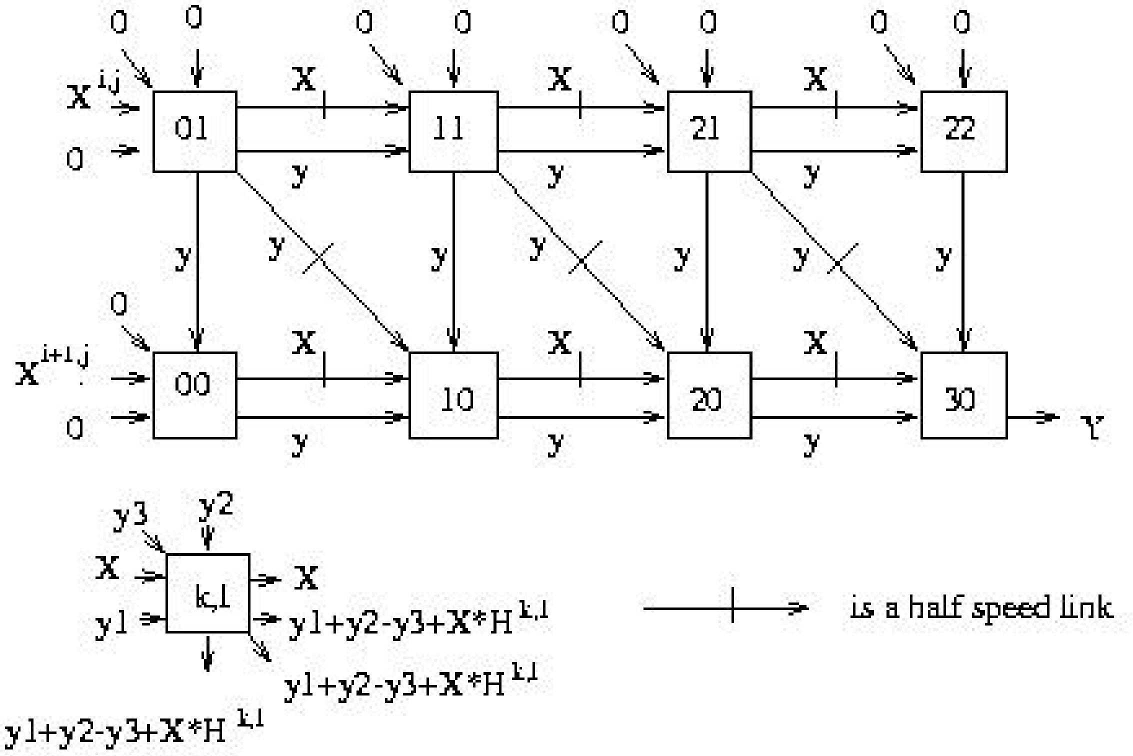

Figure 4. Processor array for 2D convolution. K = 2, L = 4

Data propagates through the array at different speeds. Note the *X* links and the diagonal *y3* links are half the rate of the horizontal and vertical *y* links. A report of the simulation and verification of the array can be found in (Dunstan, 1998). The array may be used to compute convolution (equation 2.1) and therefore may be used in the computation of normalized correlation (equation 4.3).

The same network can be used to directly compute correlation by equations 4.1 and 4.2 by modifying the cell operation to correspond to either sum of absolute differences or sum of square of differences operation. That is, the cells produce either $y1 + y2 - y3 + |X - H|$ or $y1 + y2 - y3 + (X - H)^2$. For instance, Figure 5 shows the required cell operation for the processor array to compute equation 4.1.

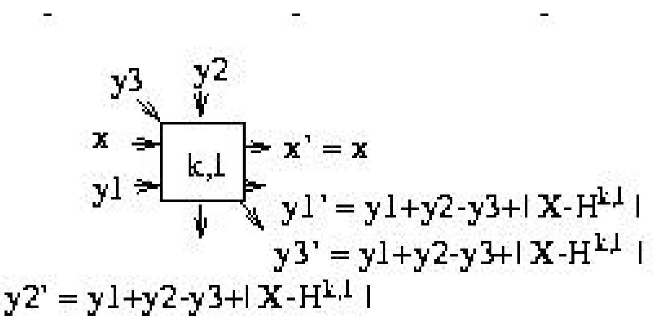
y3
y2
x
y1
k,l
x' = x
y1' = y1+y2-y3+| X-H^{k,l} |
y3' = y1+y2-y3+| X-H^{k,l} |
y2' = y1+y2-y3+| X-H^{k,l} |

Figure 5. Cell operations corresponding to equation 2.1

6. FPGA Implementations

In the last section, a systolic array to support object detection was described. In order that a device based on this design is to be useful in general circumstances, it should be able to accommodate applications requiring different template weights and sizes. A recent enabling technology for special-purpose devices is Field Programmable Gate Arrays (Oldfield, 1995). FPGAs were once used to prototype algorithm-specific integrated circuits and now have the capacity and speed to serve as end products. In this section, FPGA-based implementations of the processor array are considered for each of equations 2.1, 4.1 and 4.2. Although the same array structure may be used, the operation of the processing cells will differ for each equation. The essential difference is that equation 2.1 requires constant-value multiplication; equation 4.1 requires no multiplication and equation 4.2 requires general-value multiplication.

Constant-value multiplication can be efficiently implemented using distributed arithmetic and lookup tables (Burrus, 1977) instead of circuitry-intensive general-value multiplication. Possible products of the template weights (the constants) are preloaded into a lookup table. Distributed arithmetic may be used for a processing cell designed for equation 2.1 as follows. The *X* value coming in to a processing cell can be decomposed into four 4-bit nibbles. Each nibble is used to select from a lookup table of 16 preloaded possible partial products of the constant value *H*. The four selected partial products are added to produce the whole product.

Equation 4.2 requires general-value multiplication. Wojko and ElGindy (1997) investigated multiplication circuitry in FPGAs and found that parallel adder multipliers for 16-bit numbers consumed approximately 50% or more of logic cells in Altera FLEX81188 and Xilinx XC6216 FPGA chips. Hence a fast implementation of the processor array for executing equation 4.2 in these FPGAs would require one FPGA per processing cell.

By contrast, a processing cell corresponding to equation 4.1, which requires no multiplication, takes around 320 logic cells in a Xilinx XC6216 allowing an array of possibly 12 processing cells (perhaps a 3 × 4 template) to be housed in a single chip. The digital logic required for the processing cell is shown in Figure 6. The registers shown are in order to slow the propagation rate of those data streams although separate clock rates may be used instead. Additional registers for each output would be required for interfacing to other cells in the array.

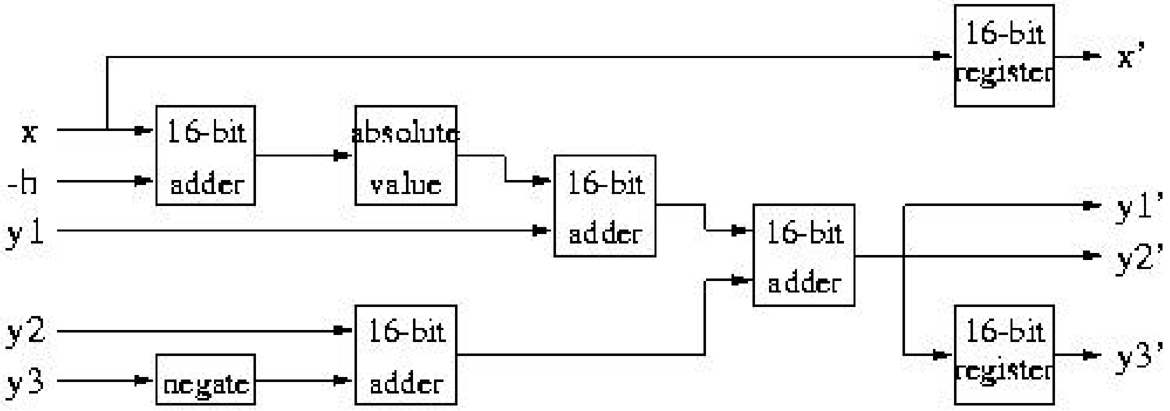

Figure 6. Processor array cell logic corresponding to equation 4.1

The current generation Xilinx Virtex FPGA chip has up to 25,000 logic cells, permitting a possible array of 72 cells of the type required for equation 4.1 (perhaps an 8×9 template). An array using general-value multiplication could by as large as 10 cells in a single Xilinx Virtex FPGA. The estimated template size for each type of FPGA is summarised in Table 1. Note that placement and routing constraints on the FPGA chip may reduce the actual template size that may be accommodated.

FPGA Chips	FLEX81188	XC6216	VIRTEX
Maximum # logic cells	1008	4096	25000
# Processing cells for equation 4.1	3	12	72
# Processing cells for equation 4.2	1	1	10
# Processing cells for equation 2.1	1	3	20

Table 1. Estimated maximum template sizes

7. Conclusions

Parallel processing methods can be effectively applied to the detection and classification of UXO. Specialized pipelined filters can act as hardware accelerators by speeding up the processing of high volume, low level processing tasks. A systolic processor array design was presented which can act as a two dimensional filter for computing the convolution of a data set and an object's signature response. In this form, the processor array can support normalised correlation by executing a computationally intensive component. Moreover, the same processor array structure, with different processing cell operations, can directly compute simpler correlation methods. Reconfigurable hardware is appropriate for implementing the processor array because it can be adapted for different templates corresponding to signature responses for different objects, orientations and depths. Possible implementation of the processor array in Field Programmable Gate Arrays was discussed, including estimates of the maximum template sizes that could be accommodated in current generation FPGA chips.

REFERENCES

P. M. Athnas and A. L. Abbott, ``Real-Time Image Processing on a Custom Computing Platform'', *IEEE Computer*, February, 1995.

J. M. Arnold, D. A. Bell and E. G. Davis, ``Splash-2", *Proc. Fourth Annual ACM Symposium on Parallel Algorithms and Architectures*, 1992.

C. S. Burrus, ``Digital Filter Structures described by Distributed Arithmetic'', *IEEE Transactions on Circuits and Systems*, 1977, pp 674-680.

P. J. Clark, ``Quality Assurance of Digital Sub-Surface UXO Detection", *Proc. UXO Forum 98*, 1998.

N. Dunstan and P. M. Lenders, ``Systolic Arrays for 2D Decimation Filters", *Proc. Second International Conference on Algorithms and Architectures for Parallel Processing,* Singapore, June, 1996, pp 14-20.

N. Dunstan, ``PIP: A Parallel Processing System in PVM", *Proc. Fifth Australian Conference on Parallel and Real-Time Systems,* Adelaide, 1998, pp 332-342.

H. T. Kung, ``Why Systolic Architectures", *Computer*, Vol. 15, No. 1, January, 1982, pp 37-46.

J. V. Oldfield, *Field Programmable Gate Arrays: Reconfigurable Logic for Rapid Prototyping and Implementation of Digital Systems*, John Wiley and Sons, 1995.

R. J. Petersen and B. L. Hutchings, ``An Assessment of the Suitability of FPGA-based Systems for Use in Digital Signal Processing'', *Proc. Fifth International Workshop on Field Programmable Logic,* Oxford, August, 1995

M. Wojko and H. ElGindy, ``Comparitive Analysis of Multiplication Techniques for FPGA Architectures'', *Proc. Fourth Australian Conference on Parallel and Real-Time Systems,* Newcastle, 1997, pp 446-457.